Predicting hospitalization for patients

Research Paper Business Analytics

January 2013

Mohamed Yerou

Supervisor: dr. Mark Hoogendoorn

VU University Amsterdam
Faculty of Sciences
De Boelelaan 1081
1081HV Amsterdam
The Netherlands

Preface

This paper is part of the curriculum of the master program Business Analytics at the VU University in Amsterdam. The purpose of the paper is to do research on a subject that emphasizes the business-related aspects of the program as well as the more fundamental aspects of mathematics and computer science. The input for this research is drawn form an existing patient data provided by The Heritage Provider Network (HPN)[1] to predict and prevent unnecessary hospitalizations.

The main purpose of this paper is to create an algorithm that predicts how many days a patient will spend in a hospital in the next year using available historical claims data from previous years in order to prevent unnecessary hospitalizations.

I would like to thank my supervisor, Mark Hoogendoorn, for his support during the whole process of research and writing this paper.

Contents

1. Introduction

The Heritage Provider Network Health Prize is a competition to develop a breakthrough algorithm that uses available patient data to predict and unnecessary hospitalizations by identify those patients most likely to be admitted to hospital. The competition is run for two years, with milestone prizes awarded every six months.

The essential purpose of this paper is not to win the Heritage Health Prize, but to do some exploratory data analysis (EDA) on the provided patient data and to use mathematics, computer science and data mining techniques to develop a mathematical model that will be able to identify which patients will end up in the hospital.

Many healthcare organizations have developed a large range of numerical and simulation techniques that are applied in daily decision-making. Although the application of these techniques does not appear as widespread as in other sectors, the organizations that have long invested in new technology infrastructure such as comprehensive business intelligence are reaping now significant business performance improvement.

The purpose of this paper is to start from a real problem, which is the high expenses spent on unnecessary hospital admissions, and then seeking a modeling solution for it that uses the available patient data to predict and prevent unnecessary hospitalizations in a form that can help to inform decisions about clinical practices and health-care resource allocations. The main question to be answered is:

Can we predict the number of days that a patient will spend in the hospital using available historical claims data from previous years?

In this paper I try to model the number of days that the members will spend in a hospital using the following data mining techniques: radial-basis function (RBF), multilayer perception (MLP), multiple linear regression and classification decision tree.

The outline of this paper is as follows. First of all, a short data description is given to be able to process and understand the data context, where several data manipulations and transformations are taking place. Then a short description of the models to use is followed. Finally, the answer to the research question is formulated by describing and applying the built models.

2. Dataset

Heritage Provider Network (HPN) provided three zip files containing the historical claims data needed to work on, namely HHP_release3.zip. A summary of the fields descriptions of the data provided is given in the following table.

Description of Data Fields based on the data description on http://www.heritagehealthprize.com/c/hhp/data

Dataset	Field	Description
Members Table	MemberID	A unique member ID
	AgeAtFirstClaim	Member's age when first claim was made in the Data Set period
	Sex	Male or female
Claims Table	MemberID	A unique member ID
	ProviderID	The ID of the doctor or specialist providing the service
	Vendor	The company that issues the bill
	PCP	member's primary care physician
	Year	The year of the claim, Y1, Y2, Y3

	Specialty	
	PlaceSvc	Place where the member was treated
	PayDelay	The delay between the claim and the day the claim was paid for
	LengthOfStay	Length of stay in hospital
	DSFS	Days since first service that year
	PrimaryConditionGroup	A generalization of the primary diagnosis codes
	CharlsonIndex	A generalization of the diagnosis codes in the form of a categorized comorbidity score
	ProcedureGroup	A generalization of the CPT code or treatment code
	SupLOS	A flag that indicates if LengthOfStay is null because it has been suppressed
DaysInHospital Tables	MemberID	A unique member ID
	ClaimsTruncated	A flag for members who have had claims suppressed. If the flag is 1 for member xxx in DaysInHospital_Y2, some claims for member xxx will have been suppressed in Y1
	DaysInHospital	The number of days in hospital Y2 or Y3, as applicable

The main dataset is the Claims Table which contains historical member"s claims from 3 years, Y1, Y2, Y3. This dataset contains 2668990 records which are patients" claims during the three years. To use this dataset we need to understand it and maybe make some data manipulations by creating, modifying or adding new features. Moreover, we need to separate it into a training data set and a testing data set. How to choose the training dataset and testing dataset depends on which model we want to build and the methods we will use, which I will describe in the following paragraph.

3. Data understanding

Data understanding is an important step in data analyzing tasks, as it can help getting an idea how to start and which techniques and methods to use. Furthermore, it can also help making some decisions from the beginning how to predict the outcomes.

Since we deal here with a hospital data, Heritage Provider Network published less information to achieve the goal of the competition due to the issue of protecting individual patient privacy as announced in the HIPAA Privacy Rule. That"s why a lot of time was needed to make assumptions. To understand the structure of the data and the variables, I was in this step more or less “playing around” and not fully aware of the complexity of problem, however the more you play with it the more you could understand about the aim and the possible steps to follow. To do some exploratory data analysis I have used RapidMiner, which has the ability to treat data of big size, to check and look if there are some relations between the attributes, if there are outliers and whether there is any missing value.

The statistical results of the claim table ware:

Claim
Data Table
Number of examples = 2668990
14 attributes:

Name	Type	Range	Missings
MemberID	integer	=[4 - 99998824]	= 0
ProviderID	integer	=[472 - 9999241]	= 16264
Vendor	integer	=[39 - 999874]	= 24856
PCP	integer	=[74 - 99905]	= 7492
Year	polynominal	=[Y1, Y2, Y3]	= 0
Specialty	polynominal	=[Anesthesiology, Diagnostic Imaging, Emergency, General Practice, Internal, Laboratory, Obstet...	= 8405
PlaceSvc	polynominal	=[Ambulance, Home, Independent Lab, Inpatient Hospital, Office, Other, Outpatient Hospital, Urg...	= 7632
PayDelay	polynominal	=[0, 1, 10, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 11, 110, 111, 112, 113, 114, 115, 11...	= 0
LengthOfStay	binominal	=[1 day, 2- 4 weeks]	= 2611333
DSFS	polynominal	=[0- 1 month, 1- 2 months, 10-11 months, 11-12 months, 2- 3 months, 3- 4 months, 4- 5 months...	= 52770
PrimaryConditionGroup	polynominal	=[AMI, APPCHOL, ARTHSPIN, CANCRA, CANCRB, CANCRM, CATAST, CHF, COPD, FLaELEC, F...	= 11410
CharlsonIndex	polynominal	=[0, 1-2, 3-4, 5+]	= 0
ProcedureGroup	polynominal	=[ANES, EM, MED, PL, RAD, SAS, SCS, SDS, SEOA, SGS, SIS, SMCD, SMS, SNS, SO, SRS, SUS]	= 3675
SupLOS	integer	=[0 - 1]	= 0

This table gives us a statistical view of the most important dataset, the Claims Table, which contains historical member"s claims from 3 years together. As we can see there are 14 attributes, most of them are not numeric (nominal), which will make it a bit difficult to model the problem since some of the models I will use are dealing only with numerical data.

In addition there are 2668990 claims in the Claims table. This Claims table and the target tables are separated; moreover there are two target tables: a DaysInHospital_Y2 table contains members who made a claim in Y1 and were eligible to make a claim in Y2 and a DaysInHospital_Y3 table contains members who made a claim in Y2 and were eligible to make a claim in Y3. Similarly, target.csv file contains members who made a claim in Y3 and were eligible to make a claim in Y4. To be eligible means to be an HPN member (regardless of whether or not a claim was made.)

In other words, the target of a specific claim in year 1 will be the number of days spend in hospital in year 2, and the target of a specific claim in year 2 will be the number of days spend in hospital in year 3. We are given only two tables including the number of days spent in hospital: one contains DaysInHospital of year 2 and the other containing the DaysInHospital of year 3. This means that the targets of the claims from the third year, which will be the number of days spend in hospital in year 4, have to be predicted and that is exactly what I"m trying to do here in this research paper. To do this I have to separate the Claims table first into three sub datasets each containing claims of one specific year: year1, year2 and year3 (see the Java code in Appendix A.) Then, each sub dataset will be merged into the corresponding target vector of the corresponding year after doing possible transformation on the subset data. I have done this task using Visual Basic for Applications (VBA) in Excel (see VBA code in Appendix B).

Moreover, another separated dataset which have to be merged into the obtained subsets is the Members datase. This dataset contains three attributes, MemberID, ageAtFirstClaim (member's age when first claim was made in the Data Set period) and Sex. The idea is to assign, for each member/year in the each subsets, the corresponding age and sex category. I have done this using VBA in Excel (see VBA code in Appendix B),

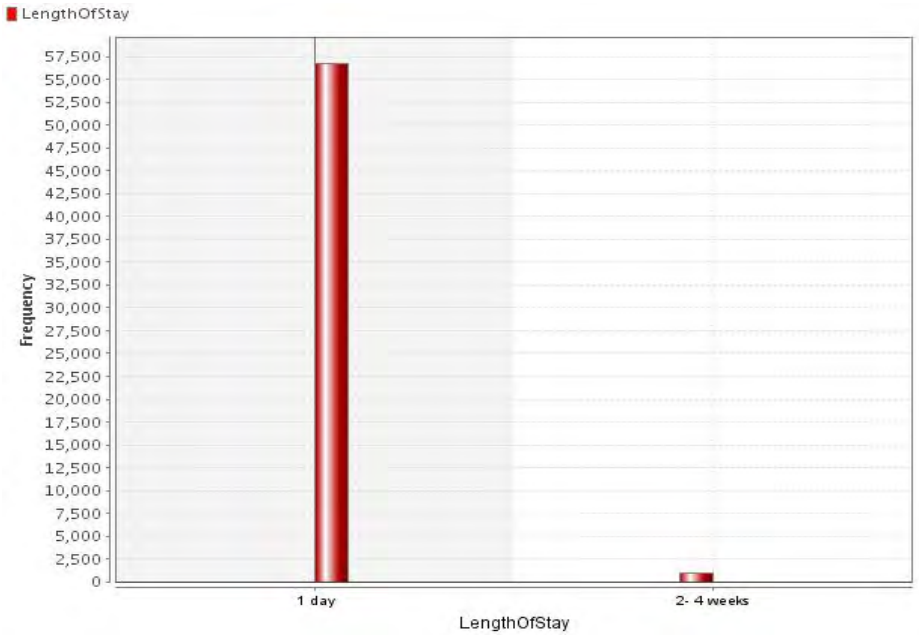
3.1. Missing values

Missing data is a common problem in statistical analysis. There are several methods that have been proposed in the literature to treat missing data which can be divided into three categories [2, 3]: a) Case/Pairwise Deletion, b) Parameter estimation and c) Imputation techniques. There are also traditional alternatives for handling missing data like: Comparing the missing and non-missing cases on variables where information is not missing, dropping variables or dropping subjects, i.e. case-wise deletion of missing data. One of the more frequently used is treating missing values just as another new category. The interest in dealing with missing values is not only applied to the statistical applications but also to new areas such as Data Mining [4] and Microarrays [5, 6]. These applications include supervised classification as well as unsupervised classification (clustering). In microarrays data some people even replace missing values by zero. The most of methods are dealing with numerical values in the dataset, which means that the treatment of missing values that are not numerical a bit difficult, which is the case of our dataset where the most of attribute are not numeric (nominal). One method to deal with nominal variables is to treat missing data as just another category. To deal with the missing values in our dataset, I have tried to apply the common used methods starting by analyzing the nature of the attributes containing missing values as follows.

As we can see from the statistical table above the Claims table contains 14 attributes, 9 of this attributes are not numeric and 5 are integers. Some of this attributes have a lot of missing values and some not. The first attribute in the table is **MemberID:** this pseudonyms is used as the key identifier for each patient based on the generation of sets of random numbers that were then assigned randomly during the generation of the data that why it has no missing value. Furthermore it will be not useful for learning as it does not generalize.

LngthOfStay:

Another remarkable attributes is the attribute LngthOfStay which has 2611333 missing values (almost 98% missing values). The question here was if these missing values simply mean no information, and how to deal with this kind of missing information? To answer this question we take a look at the distribution of this attributes in the following histogram.



As we can see there are only two values allowed to this attribute, "1" and "2-4weeks". There are almost 57500 records where LengthOfStay takes value “1” and almost 1000 records where it takes value “2-4weeks” and rest which is almost 2668990-58500=2610490 are blank.

The standard definition of the inpatient record is that patient stayed in the hospital overnight, according to HPN, this means that LengthOfStay for every inpatient record should be at least 1 day. Unfortunately, there are thousands of records where inpatient stays do not have any LengthOfStay.

Another attribute that may help understanding why LngthOfStay has so many missing values is the attribute SupLOS which indicates if the blank for the LengthOfStay variable is due to suppression done during the de-identification process. According to HPN, if there is a blank LengthOfStay and SupLOS is 0, then this is how it was when it came out of the HPN dataset. If there is a blank LengthOfStay and SupLOS is 1, then LengthOfStay has been suppressed. For this reason there are so many missing values of the LngthOfStay attribute due to suppressions for privacy reasons and lack of information. From the literature [3], the easiest and more commonly applied method in this case is to drop the variable with a substantial proportion of cases with missing data. For this reason I have decided to delete this attribute from the dataset.

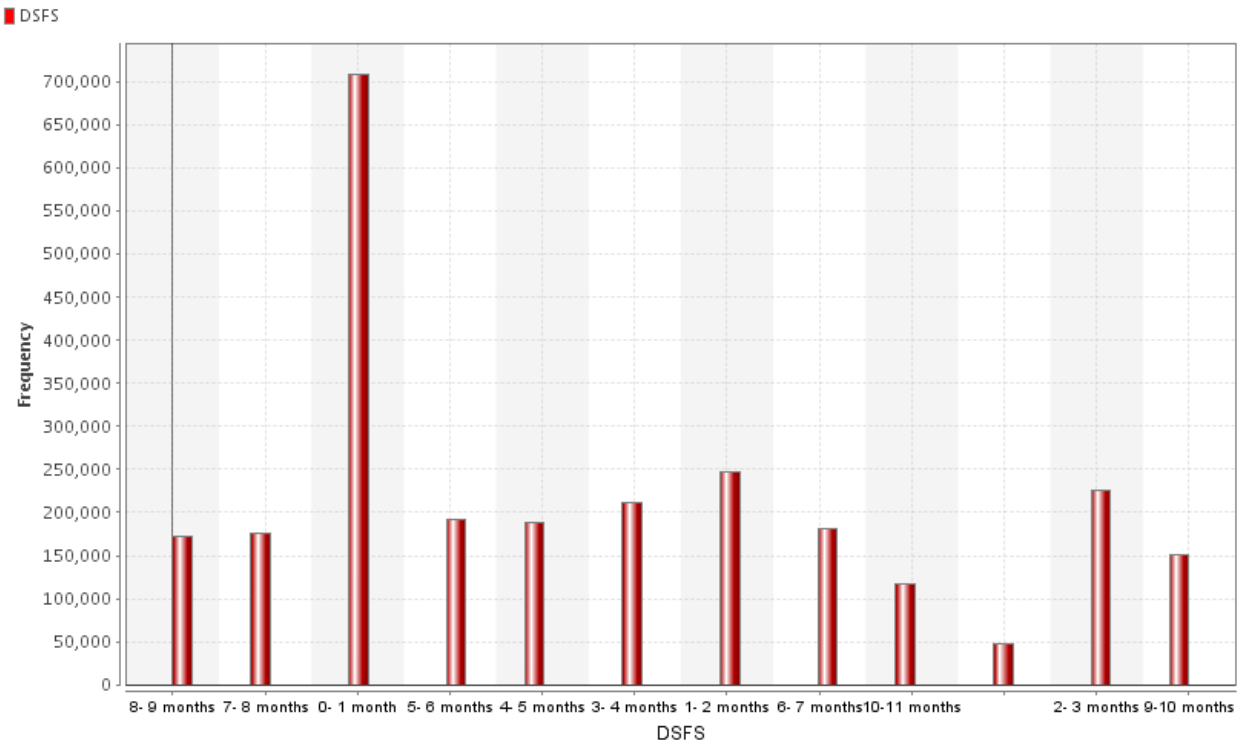
ProviderID, Vendor and PCP:

There are also three other attributes with many missing values which are: ProviderID, Vendor and PCP. According to HPN, these pseudonyms are all used for identification variables based on the generation of sets of random numbers that were then assigned randomly during the generation of the data. This means that doing some analysis on this attributes only may not help a lot in predicting how many days a patient will stay in hospital.

According to HPN, the existence of these missing values in these three attributes means only that the patients had no vender, no provider or no primary care physician (PCP). But since having vender, provider or PCP may have some influence on the number of days that will be spent in the hospital in the future; I decided to apply one of the common used methods, which is treating this missing data as just a new category.

DSFS (days since first service that year):

As the table above shows, this attribute contains many missing values which are all blanks. Since DSFS is the time since that customer's first claim, it is expected that every MemberID in the Claims file has at least a "0- 1 month" entry for DSFS. That's why these blank fields are considered as missing information according to HPN. To deal with this missing data, we take first a look at the following DSFS histogram.



From this histogram we can see that the majority of DSFS are with value: "0-1 month", but some are different. One idea is to replace the missing values for the DSFS attributes by the most common value for this attribute which are "0-1 month". But since this may causes loss of information that can have influence on the number of days that will be spent in hospital, I have chosen the common used method as before treated this missing data as just new category “no-month”.

Specialty, PlaceSvc (PS), ProcedureGroup (PG) and PrimaryConditionGroup (PCG):

These attributes are all polynomial as the statistical table shows. They are all containing missing values. To deal with these missing values I have used the same as for previous one, which is treating the missing data as just a new category. Thus for the attributes “Specialty”, “PS”, “PG” and “PCG” I have assigned the following new categories for the missing fields respectively, “Unknown-Specialty”, “Unknown-PS”, “Unknown-PG” and “Unknown-PCG”.

Let us now consider the following statistical results of the Members dataset.

members
Data Table
Number of examples = 113000
3 attributes:

Name	Type	Range	Missings
MemberID	integer	=[4 - 99998824]	= 0
AgeAtFirstClaim	polynominal	=[0-9, 10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80+]	= 5753
Sex	binominal	=[F, M]	= 17552

From this table we can see obviously that there are many cases missing for both variables, AgeAtFirstClaim and Sex. One idea how to treat these missing values is to assign them randomly, but doing so may cause loss of information since we don't know its relevance to the number of days that will be spent in hospital. Therefore it is not a good approach to assign them randomly. That why I have decided to consider the blanks in both AgeAtFirstClaim and Sex variables as unknown categories: "UnknownAge" and "UnknownSex".

For all treatment of the missing values I have used Visual Basic for Applications (VBA) in Excel (see VBA code in Appendix B).

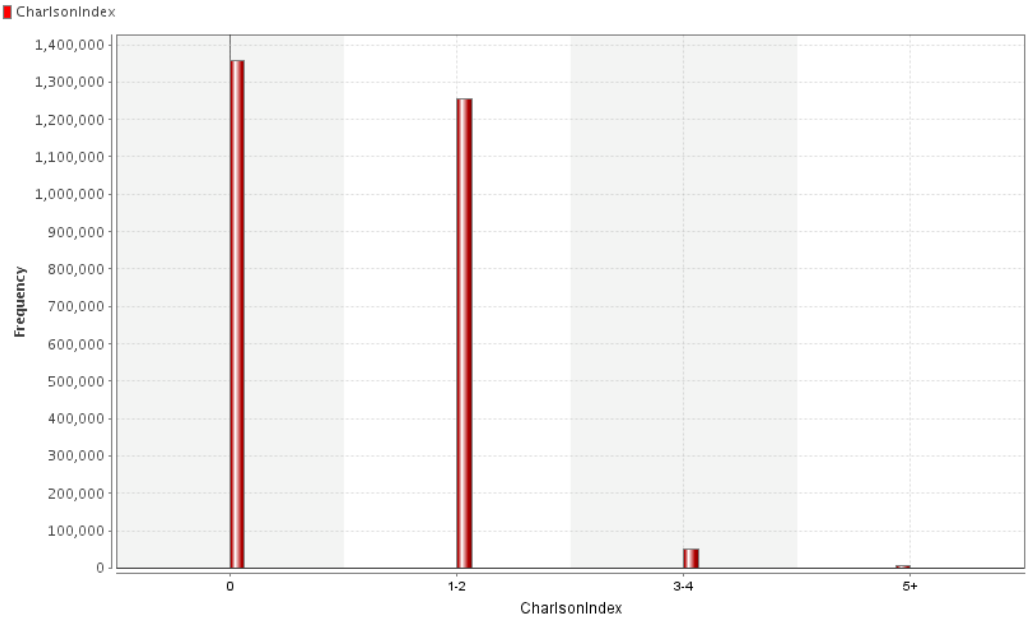
3.2. Outliers

Outliers are values in the dataset that are very different from the data values for the majority of cases in the data set or the cases with rare instances in the dataset. Identifying outliers is very important since they can influence the results of data analysis dramatically. Whether we have to delete the outliers from a dataset or not depends on the reason why the case is an outlier and what the purpose of the analysis is.

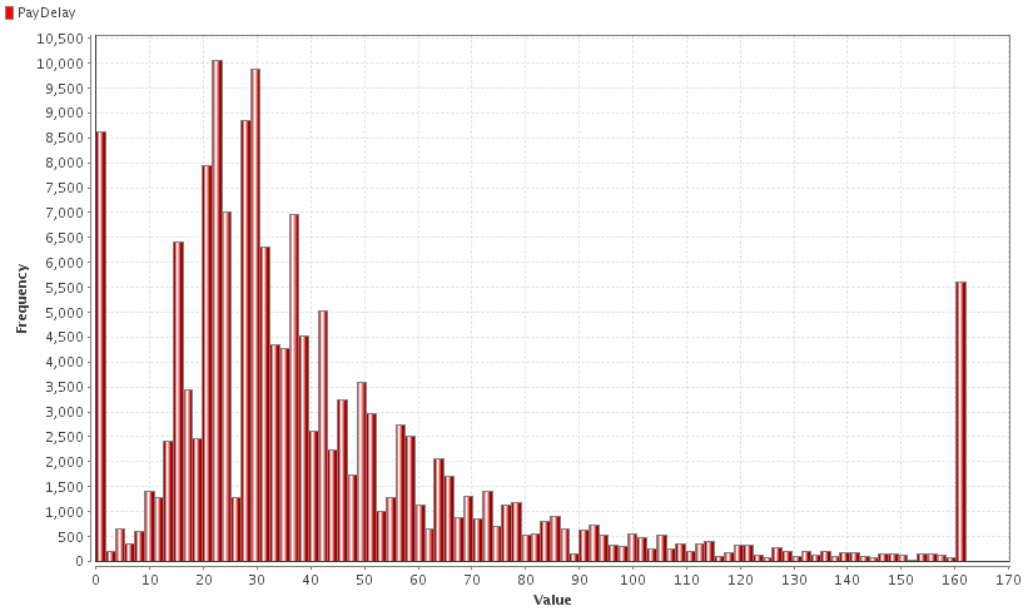
Just like missing values finding the rare instances or the outliers is also a common problem in statistical analysis, it is very important in many KDD (knowledge discovery and data-mining) applications, such as detecting credit card fraud or finding irregularities in gene expressions. There are also several methods that have been proposed in the literature [7] to deal with outliers in datasets. These methods usually make assumptions about data distribution, statistical distribution parameters, type or number of outliers. There are also other methods like depth-based method which organizes the data points in layers in the data space according to the value of the point depth. Because of the nature of our dataset I used assumptions about data distribution to determine the outliers.

Looking at different graphs of the distribution of the attributes in our dataset I have distinguished many cases which can considered as outliers, they are as follows.

CharlsonIndex [8 9]: This attribute is about the Charlson co-morbidity index which is a predictive measure of the ten-year mortality for a patient. If a patient has a range of co-morbid conditions, like heart disease, AIDS, or cancer then depending on the risk of dying associated with each condition a score of 1, 2, 3, or 6, will be assigned. Scores are then summed to provide a total score to predict mortality. From the histogram hereunder it is obvious that there are three important categories which are, "0", "1-2" and "3-4". The most claims have values in these categories, only one category is strange which is the "5+" category, according to the HPN this category means that CharlsonIndex takes values greater or equal to 5. But this category takes only a few values compering to the other categories (almost 0.2%). Since we don't know how this category will influence the prediction results and since it may have potential predictive capabilities I have decided not to remove it and to leave it as it is.



Paydelay: another example is the Paydelay where we can see two rare instances as the following histogram shows.



If we look at the distribution of this variable, then we see two significant spikes one at zero days and another at 162+ days. According to HPN these are not outliers but they are two important instances in the dataset, the first one represents claims that have zero delay and are paid immediately, the other one represents claims that are not paid or are paid after 162 days. As we don't know how this will affect the predictive model performance we want to build, I chose to consider all the variables and if we chose this attribute as a potential predictors in our model then we have to take this two particular instances in account.

4. Detailed task description

To achieve the goal of developing a model that uses available patient data to predict and prevent unnecessary hospitalizations, the competition was designed to use de data from years 1, 2 and 3 to predict how many days a patient will spend in a hospital in the year 4. Since Claims Table contains no information about the number of days of hospitalization in year 4 and there are only two separated tables which containing this information, that is the DaysInHospital Tables Y2 and Y3 which including the number of days of hospitalization for each eligible member during Y2 and Y3. The task is thus to use the relationship between Year 1 claims and Year 2 hospitalization, and Year 2 claims and Year 3 hospitalization, given Year 3 claims, we have to predict Year 4 hospitalizations.

This means that the task is to split first the Claims Table, which contains claims of the three years together, into three separated tables each contains only claims of one specific year, Year1Claimdata, Year2Claimdata and Year3Claimdata (see the Java code in Appendix A), and then, depending on the methods and models to build one can merge the table Year2Claimdata with DaysInHospital_Y3 to create a training dataset and next to use Year3Claimdata to predict DaysInHospital_Y4, or merging the Year1Claimdata with DaysInHospital_Y2 and Year2Claimdata with to create a training dataset and next to use Year3Claimdata to predict DaysInHospital_Y4.

5. Data transformation

In most cases people chose to have data entered in a format that is easy for typists (to reduce data-entry errors) but which differs from the form needed for analysis. Therefore data transformation is sometimes needed to get a new data set from an existing data set before you can begin analysis.
There are also several ways that have been proposed in the literature to transform data, one way is to calculate a new value as a direct function of existing variables in your data set [10]. Because of the nature of models I will build, where some of these models deal only with numerical data, I have decided to use one common technique from the literature [16] that is the "Categorical representation" which uses dummy variables with values 0 or 1 to transform all categorical attributes into numeric.

First of all and before starting doing possible transformation on the data I have split the original dataset (the Claims Table), which contains claims of the three years mixed together, into three separated tables each contains only claims of one specific year, Year1Claimdata, Year2Claimdata and Year3Claimdata (see the Java code in Appendix A).

Then for all attributes of the claim dataset and attributes of the Members dataset, all categories are translated to separate columns (new generated categories), where for each member/year these columns contain the number of appearances of that category in the original table. To do this, I have used the reference cell coding model (Kleinbaum et al., 1998) which assigns for given claim, for each new category (column) of each attribute in the original dataset, number 1 if this claim contains this category and 0 if not. I have done this transformation using VBA in Excel (see VBA code in Appendix B). The following table shows the transformed attributes and number of the categories that are transformed for each attribute.

Attributes	Number of categories that are transformed
Specialty	13
PlaceSvc	9
DSFS	13
PayDelay	7
CharlsonIndex	4
ProcedureGroup	18
PrimaryConditionGroup	46
AgeAtFirstClaim	10
Sex	3

After doing this transformation we get a new datasets, where each member has 123 attributes with “0” or “1” as value. But because members may have multiple claims in the same year and the days in hospital are assigned to members and not to claim, summing up all the claims of one member in one record was needed to do predictions on member en not on claims. That"s why, for each member I have add up the values the 123 attributes of each claim of this member to form one record for each member. The new dataset obtained has now values different than “0” or “1”.

6. Modeling and Evaluation

The objective function for the model to build will based on the degree of accuracy of its predictions, which will be judged by comparing the predicted number of days a member will spend in the hospital and the actual number of days this member spent in the hospital. As subject to the Heritage Health Prize Competition Official Rules the prediction accuracy has to be calculated based on the following evaluation error metric which is given on http://www.heritagehealthprize.com/c/hhp/details/evaluation:

$$\varepsilon = \sqrt{\frac{1}{n}\sum_{i}^{n}[\log(p_i + 1) - \log(a_i + 1)]^2}$$

Where i = 1…n a member, n is the total number of members, p_i is the predicted number of days spent in hospital for member i in the test period, and a_i is the actual number of days spent in hospital for member i in the test period.

6.1. Models description

Since we deal here with a complex enough classification problem, I have decided to use data mining techniques to predict how many days a patient will spend in a hospital in the year 4. Especially I have illustrated four models namely, radial-basis function (RBF), multilayer perception (MLP), multiple linear regression and classification decision tree. Hereunder follows a short description of each model. I will also try to use the principle component analysis technique which helps to identifying patterns by re-dimensioning and expressing the data in such a way that can improve the performance of the best chosen model.

6.1.1. RBF network model

Radial basis function (RBF) network is one of the artificial neural network that uses radial basis functions as activation functions. It is a feed-forward network which uses both supervised and unsupervised training algorithms to find the parameters of the network.

Radial basis function (RBF) networks typically have three layers: the first layer is the input layer which is made up of source nodes whose, the second layer is the hidden layer with a non-linear RBF activation function connected directly to all of the nodes in the input layer and finally a linear output layer which is a linear combination of radial basis functions of the inputs and neuron parameters.

The following figure gives the architecture of a radial basis function network. An input vector x is used as input to all radial basis functions, each with different parameters. The output of the network is a linear combination of the outputs from radial basis functions which is as follows, $y = \sum_{i=1}^{m} w_i \varphi(\| x - t_i \|)$ where $\| x - t \|$ distance of x= (x_1 ,..., x_n) from vector t. The norm is typically taken to be the Euclidean distance and the radial basis function is taken to be Gaussian as follows $\varphi(r) = \exp(-\frac{r^2}{2\sigma^2})$.

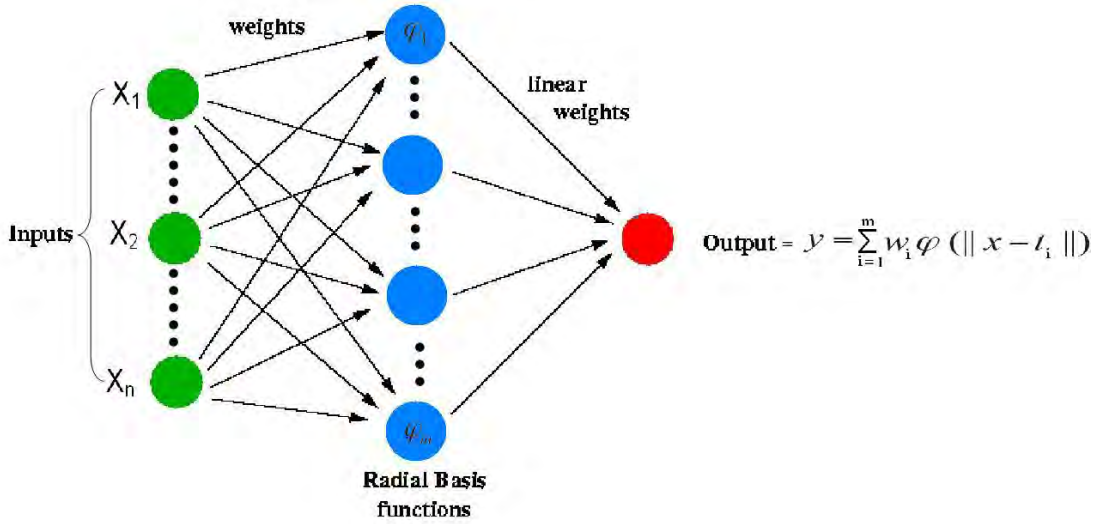


Figure 1: Schematic diagram of the architecture of a radial basis function network from http://www.cenaero.be/Page.asp?docid=27097

When a radial-basis function (RBF) network is used to perform a complex pattern classification task, the problem is basically solved by transforming it into a high-dimensional space in a nonlinear manner. The underlying justification is found in Cover's theorem on the separability of patterns, which, in qualitative terms, may be stated as follows:

A complex pattern-classification problem cast in a high-dimensional space nonlinearly is more likely to be linearly separable than in a low-dimensional space [14].

6.1.2. MLP netwerk model

A multilayer perceptron (MLP) is also one of the artificial neural networks that use multiple layers of nodes, where each node is a neuron with a nonlinear activation function. MLP is also a feed-forward network which uses a supervised training algorithm called backpropagation for training the network.

The following figure gives the architecture of a multilayer perceptron network. The MLP network uses an input vector x to computes a single output by forming a linear combination according to its input weights and then using a n activation function as follows to compute the output y: $y=\varphi(net) = \varphi(w_0+x_1w_1+\ldots x_nw_n)$ where φ is sigmoid or logistic function.

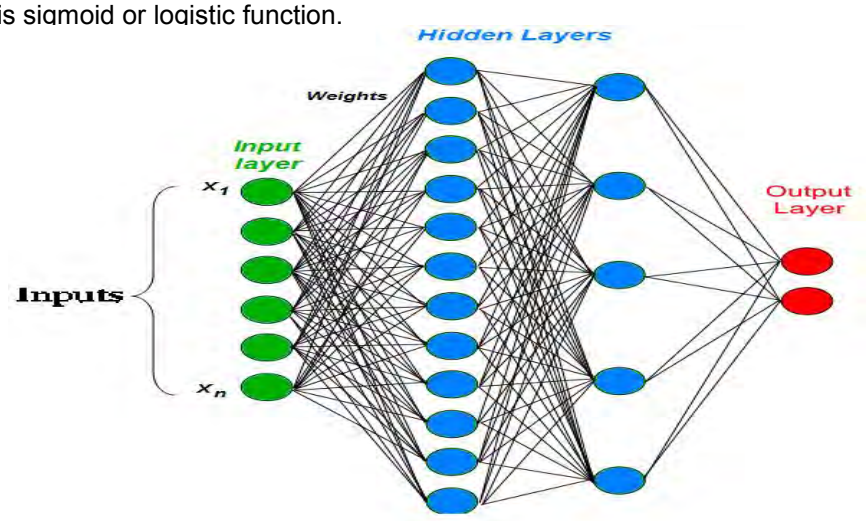


Figure 2: Schematic diagram of the architecture of a multilayer perceptron network from http://lefteris.realintelligence.net/?p=931

6.1.3. Multiple Lineair Regression

The multiple linear regression model is an extension of the simple linear regression model to model the relationship between two or more explanatory variables and a response variable by allowing the response variable to be a function of multiple explanatory variables $X_1,X_2,\ldots,X_n$.

Mathematically the relationship between the explanatory variables $X_1,X_2,\ldots,X_n$ and the response variable Y can be written as $Y= b_0+b_1X_1+b_2X_2+\ldots+b_nX_n$, where $b_0,b_1,\ldots,b_n$ are the regression parameters which have to be estimated by fitting the model to a given data set with corresponding response variables.

6.1.4. Classification Decision Tree

Classification Decision Tree is a very common used predictive model which has the ability to learn a classification function in order to predict the value of a target variable based on several input variables. It is a supervised classification model since the dependent attributes and the number of classes has to be given in advance to learn the model. The learning process will be repeated by splitting the source set into subsets based on an attribute value test.

The learning algorithm for the Classification Decision Tree model happens in a recursive manner called recursive partitioning. The stop condition for the recursion will occur when the subset at a given node has all the same value of the target variable or when splitting no longer adds value to the predictions.

The following figure gives the architecture of a Classification Decision Tree. Each interior node of this tree corresponds to one of the input variables; there are edges to children for each of the possible values of that input variable. Each leaf represents a value of the target variable given the values of the input variables represented by the path from the root to the leaf.

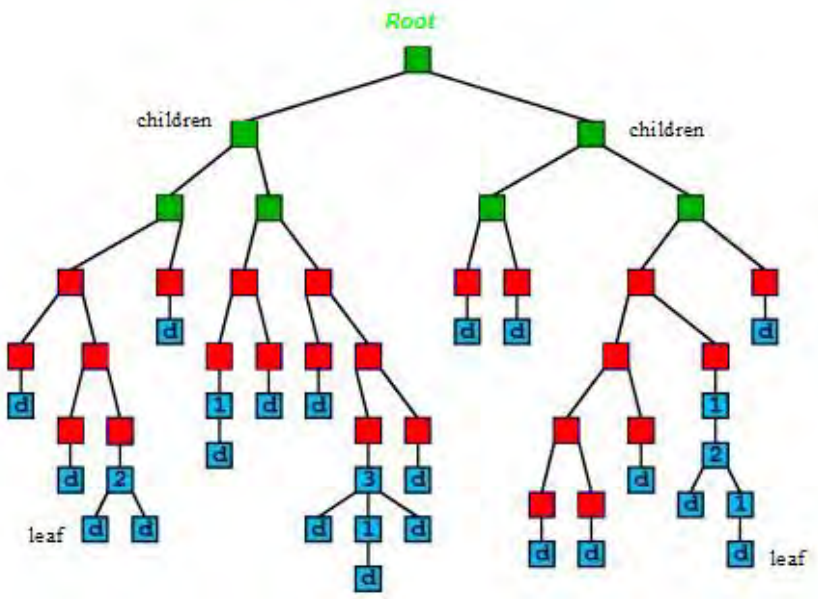


Figure 2: Schematic diagram of the architecture of a Decision Tree from
http://www.cs.uga.edu/~maria/classes/4500-Spring-2012/project3-scala.html

6.1.5. Principle component analysis

Principal Components Analysis (PCA) is a useful statistical technique that has found application in fields such as face recognition and image compression, and is a common technique for finding patterns in data of high dimension. Since patterns in data can be hard to find in data of high dimension, where the luxury of graphical representation is not available, PCA is a powerful tool for analyzing data. The other main advantage of PCA is that once you have found these patterns in the data, and you compress the data, i.e. by reducing the number of dimensions, without much loss of information [15].

To perform a Principal Components Analysis on a set of data the following steps are needed. First of all we need to calculate the covariance matrix that represents the adjusted dataset, and then we should find the eigenvectors and eigenvalues of this covariance matrix. These eigenvectors will be then ordered by eigenvalue from higher to lower to get the components in order of their significance. The idea of PCA is to ignore the components (eigenvectors) of lesser significance to project the initial data only onto the space generated by the more significant eigenvectors; we do this by multiplying the transpose of the matrix constructed by the chosen eigenvectors on the left of the transpose of initial data set. This step will produce an encoder of the original dataset. To get the old data back we have to decode the resulting encoded data by taking the matrix constructed by the chosen eigenvectors and multiply it on the left of the matrix of the encoded data.

After finding an encoder en decoder for the dataset, that is to ignore a number of less significant components (eigenvectors) which means reducing the number of dimensions, but without much loss of information. Then a distance measure is defined between the original data and the data after being encoded and decoded again. As PCA works with matrices the matrix norm is usually choose as a distance measure between two matrices, the matrix of the original data and the matrix of the data after being encoded and decoded again.

In our case I will try first to apply the three models described above to see which of them will perform better and then I will apply the PCA technique only on the best one to see if we can make some improvements of the accuracy of that model in predicting the number of days that the patient will spend in the hospital in the fourth year.

7. Prediction

To use the described models in predicting the number of days that a patient will spend in hospital, the models have to be trained first. The training step I have used is separating training/predicting tasks as follows.

According to HPN one way to use the data to do predictions is to training on Year1Claimdata using DaysInHospital_Y2 as target vector, training again on Year2Claimdata using DaysInHospital_Y3 as target vector and then using Year3Claimdata to predict DaysInHospital_Y4. This means using the claims dataset of the first year with its target vector (DaysInHospital_Y2) form year 2 and the claims dataset of the second year with its target vector (DaysInHospital_Y3) form year 3 to train a given model and then using the claims dataset of the third year and the trained model to predict the output (DaysInHospital_Y4) for year 4. For this purpose I have used the Matlab Package Netlab for all the models described above using (see Matlab code in the Appendix C).

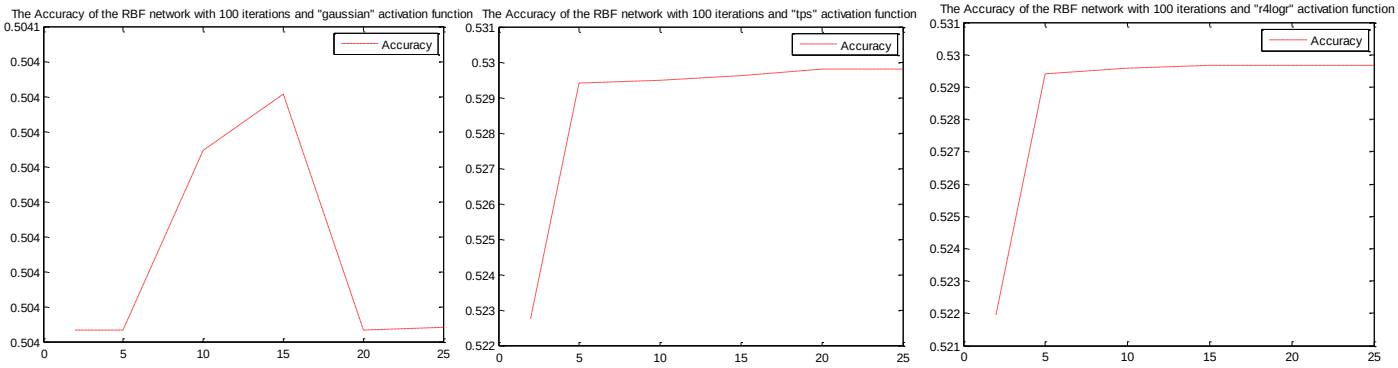
I have chosen for this approach because I used Matlab and it is not possible to load the whole original dataset in it to do the predictions that"s why separating training/predicting tasks on this way make it possible to load the subsets data in Matlab.

I will test the four models described above to see which one will perform better and give the beat accuracy. The best one will be used to do the prediction of number of days that a patient will spend in

hospital. Therefore, to create the test- and training datasets I have used 10-fold cross-validation for all the four models, and then I take the average accuracy over the two years and over the folds.

7.1. Applying the RBF network model

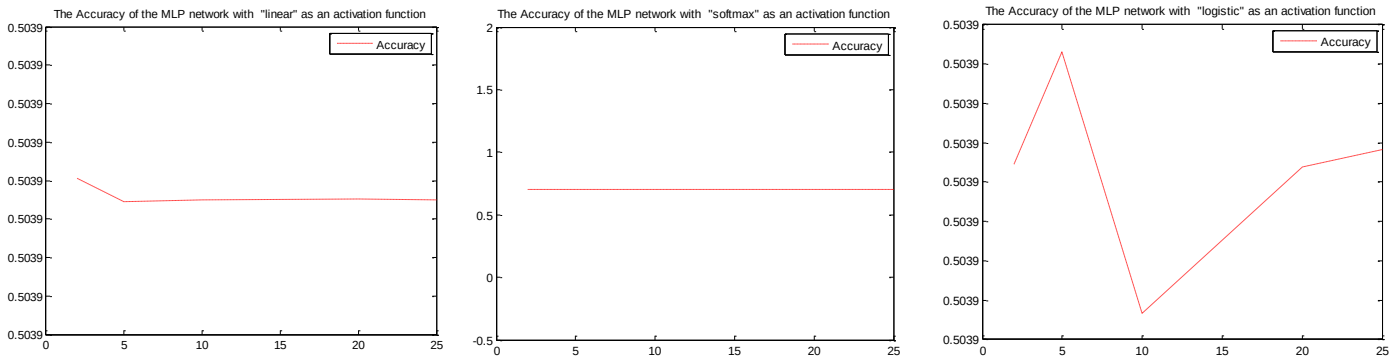
Because the accuracy of the RBF model depends on the number of kernels and the type of activation function used, I have calculated the accuracy of this model on the test-datasets for different number of kernels (2, 3, 4, 5, 7, 10, 15 and 25 kernels) and different types of activation function. Furthermore I use 10-fold cross-validation and for each number of kernels I take the average accuracy over the two years and over the folds. For the calculations I have used the Matlab Package Netlab which includes “RBF” facilities (see the Matlab code I have used to do the calculations in Appendix C). The results are as follows.



These graphs show the accuracy of the RBF model for different number of kernels and for the three possible types of activation function namely “gaussian”, “tps” and “4logr” (which is r^4*log(r)). As we can see, the accuracy of the RBF model using the Gaussian activation function is fluctuating depending on the number of the of kernels, it reaches the maximum accuracy for a number of kernels of 15 and the performance of the model becomes worse when the number of the kernels becomes high. When I used other activation functions like “tps” or “4logr”, I see that the model is performing better especially with a high number of kernels is used. That is why I have decided to using this model with the “4logr” as an activation function with a number 25 of kernels. The highest reached accuracy in this case was 0.5040.

7.2. Applying the MLP network model

The accuracy of the MLP model depends also on the number of hidden units used and the type of activation function for the output. That"s why I have calculated the accuracy of the model on the test-datasets for different number of hidden units (2, 3, 4, 5, 7, 10, 15 and 25 hidden units) and different types of activation function to see which number of hidden units and which activation function give the best accuracy. Using 10-fold cross-validation I take for each number of hidden units the average accuracy over the two years and over the folds. For this purpose I have also used for all calculations the Matlab Package Netlab which includes “MLP” facilities (see the Matlab code I have used to do the calculations in Appendix C). The results are as follows.



These plots show the accuracy of the MLP model for different number of hidden units and for three possible types of activation function for output. As we can see in the plot in the middle, the accuracy of the MLP model using the activation function 'softmax' remains stable and has surprisingly a high value of 0.6994, but if we look at the predicted target vector generated during this test we see that the model assigns for all instances of this vector the same a predicted value of 1 day, which means that the model doing things worse because this is impossible thus this is not convenient and has to be rejected. Using the “linear” activation function we see, the plot on the left that the accuracy start from a value of 0.5039 for a number of hidden units of 2, then it goes down for a number of hidden units of 5 and then remains stable. If we now look at the plot on the right using the 'logistic' activation function, we see that a higher accuracy is reached; 0.5039163 for a number of hidden units of 5, and then it decreases for a number of hidden units of 10 then it increases again when the number of hidden units becomes high but it remains smaller the that of hidden units of 5. This highest reached accuracy lead me chose the 'logistic' activation function for this model with a number of hidden units 5 to do the prediction.

7.3. Applying Multiple Lineair Regression model

To apply the Multiple Lineair Regression model I have used the Matlab function „REGRESS“ using the same claims dataset as for the other models before (dataset from the two years). Furthermore I have also used 10-fold cross-validation and calculated the accuracy of the model on this test-datasets taking the average over the two years and over the folds, a maximum accuracy of 0.4876 was reached (see the Matlab code I have used to do the calculations in Appendix C.)

7.4. Applying Classification Decision Tree model

To apply the Classification Decision Tree I have used the Matlab function „ClassificationTree.fit“ to train the model and the function „predict“ to make predictions. I have used the same claims dataset as for the other models before (dataset from the two years). Furthermore I have also used 10-fold cross-validation and calculated the accuracy of the model on this test-datasets taking the average over the two years and over the folds, a maximum accuracy of 0.5839 was reached (see the Matlab code I have used to do the calculations in Appendix C.)

The following table gives again the highest accuracy reached for each model described above.

Model	RBF	MLP	LRM	CDTree
Accuracy	0.5040	0.5039	0.4876	0.5839

As we can see from this table the Classification Decision Tree model is performing better than the other models since it has the highest accuracy.

I have then used these models to do the prediction and have submitted the predicted target vector for each model to the site of The Heritage Provider Network (HPN). The resulted scores were almost the same as in the table above and the Classification Decision Tree model has scored again the best result.

From these models I have thus chosen for the best one with the highest accuracy, which is the last one the Classification Decision Tree model, to work further with in order to make the prediction of the number of days that a patient will spend in hospital. But before doing the end predictions I was wondering if reducing the number of dimensions will help doing things better, that“s why I have applied the Principle component analysis techniques for this purpose as follows.

7.5. Applying Classification Decision Tree with using PCA

Before applying the Classification Decision Tree model using PCA, I have first applied the PCA to our dataset to determine the number of principle component that have to be used to compress the data by reducing the number of dimensions without much loss of information. I have tried the model for different number of principle components, the model still gives good results without much loss of information for a number of principle components not smaller than 50. Therefore I have chosen for a small number of principle components of 60 and then I have applied again the Classification Decision Tree model using the resulting reduced dataset. The result was a surprising, because this has improved the accuracy of Classification Decision Tree model to get a higher value of 0.609.

8. Conclusion

After applying the Classification Decision Tree model using PCA, I have trained the model using the reduced dataset with only 60 dimensions instead of 124 dimensions, and then I have run the model to predict the number of days that a patient will spend in hospital. The resulting target vector was then submitted to the site of The Heritage Provider Network (HPN), and a score of 0.634094 was reached. Comparing this result to what the others have scored until now, it is a good result since there was already three milestone prizes awarded where only a small subset of data was used and the best score was only 0.4586. To compare this result which is based on the whole dataset we have to wait until April, the deadline of the competition. But we can compare this to the actual scores on the leaderboard on The Heritage Provider Network site, where the scores are going from a minimum of 0.435583 to a maximum of 0.752297.

On the leaderboard there are many good scores, but they are marked as “Benchmark” which means they are rejected, because they are generated by submitting target files consisting of random values between 0 and 15. There are even scores greater than 1, which are also rejected because it is impossible to get such scores greater than one. Taking this remarks in account a score of 0.634094 which is close to the maximum scored on the leaderboard, means a good result. Especially if we take many factors in account, like the duration of competition, the difficulty and the nature of the dataset provided, which contains not enough information to be able to apply such powerful techniques, as well the short time period that I have to finish the paper which is, comparing to the duration of the competition, not enough to investigate all the patterns in the dataset.

Moreover, there were also three milestones awarded each 6 month, where only a small deal of the dataset was provided. The scores in those milestones were even smaller than 0.5. I know it is not convenient to compare the score I have got where I use all the data with those scores where only a deal of the data was used. But since you can expect to get better results if you use enough data to do predictions, I can conclude that my score will be a reasonable result.

I'm very satisfied about my score since it is based on one of the powerful artificial intelligence techniques, the Classification Decision Tree and the Principal Components Analysis technique which has helped to improve the accuracy of the Classification Decision Tree.

Furthermore I'm sure it is possible to reach better results than I got, but then will take more and more time to be able to take all the given information in account and to improve the model to perform more accurate.

References

[1]. http://www.heritagehealthprize.com/c/hhp

[2]. Mundfrom, D.J and Whitcomb, A. (1998). Imputing missing values: The effect on the accuracy of classification. Multiple Linear Regression Viewpoints. 25(1), 13-19.

[3]. Little, R. J. and Rubin, D.B. (2002). Statistical Analysis with Missing Data. Second Edition. John Wiley and Sons, New York..

[4]. Grzymala-Busse, J.W. and Hu, M. (2000). A Comparison of Several Approaches to Missing Attribute Values in Data Mining. In RSCTC'2000, pages 340-347.

[5]. Bello, A. L. (1995). Imputation techniques in regression analysis: Looking closely at their implementation, Computational Statistics and Data Analysis 20 45-57.
[6]. Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P. Hastie, T., Tibshirani, R., Bostein, D. and Altman, R.B. (2001). Missing value estimation methods for dna microarrays. Bioinformatics, 17(6), 520-525.
[7]. Dantong Yu, Gholamhosein Sheikholeslamiy and Aidong Zhang. Finding Outliers in Very Large Datasets. Knowledge and Information Systems (2002) 4: 387-412
[8]. http://en.wikipedia.org/wiki/Comorbidity
[9]. Charlson ME, Pompei P, Ales KL, and MacKenzie CR. A new method of classifying prognostic comorbidity in longitudinal studies: development and validation. Journal of Chronic Diseases. 2008; 40 (5): 373-383.
[10]. Jamie DeCoster. Transforming and Restructuring Data. Department of Psychology University of Alabama. May 14, 2001
[11]. http://www.analytics-magazine.org/januaryfebruary-2012/503-analytics-a-the-future-of-healthcare
[12]. Jasna Kuljis, Ray J. Paul, Lampros K. Stergioulas. Can health care benefit from modeling and simulation methods in the same way as business and manufacturing has? Department of Information Systems and Computing.
[13]. http://en.wikipedia.org/wiki/Radial_basis_function_network
[14]. Simon O. Haykin, Neural Networks and Learning Machines (3rd Edition).
[15]. Lindsay I. Smith, A tutorial on Principal Components Analysis
[16]. http://www.ats.ucla.edu/stat/sas/library/nesug98/p046.pdf

Appendix A

Splitting Data

Split the original dataset into three separated tables each contains only claims of one specific year, "Year1.csv",, "Year1.csv", and "Year1.csv".

```
import java.io.*;
import java.util.Scanner;

/**
 *
 * @author Mohamed Yerou
 */
public class Split {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
     try{
  // Open the file that is the first
  // command line parameter


  FileInputStream fstream1 = new FileInputStream("Claims.csv");

  // Get the object of DataInputStream
  DataInputStream in1 = new DataInputStream(fstream1);
  BufferedReader br1 = new BufferedReader(new InputStreamReader(in1));

  String strLine1;
  String str1=new String();
  //Read File Line By Line
  while ((strLine1 = br1.readLine()) != null)   {

      Scanner scan1=new Scanner(strLine1);
      scan1.useDelimiter(",");
      int i=1;
      while(scan1.hasNext()){
          if (i==5) {
              str1=scan1.next();

          }else{
              scan1.next();
          }
          i=i+1;
      }
      if (str1.equals("Y1")){
                  Writer output;
                  output = new BufferedWriter(new FileWriter("Year1.csv",true));
                  output.append(strLine1+"\n");
                  output.close();
      }
            if (str1.equals("Y2")){
                  Writer output;
                  output = new BufferedWriter(new FileWriter("Year2.csv",true));
                  output.append(strLine1+"\n");
                  output.close();
      }
                if (str1.equals("Y3")){
                  Writer output;
                  output = new BufferedWriter(new FileWriter("Year3.csv",true));
                  output.append(strLine1+"\n");
```

```
                    output.close();
      }
  }
  in1.close();
 }catch (Exception e){//Catch exception if any
  System.err.println("Error1: 111111" + e.getMessage());
  }
}
}
```

Appendix B

Treatment of the missing values

```
Sub MissingValues() 'Treatment of the missing values
i = 1
    While IsEmpty(Sheets("Claims").Cells(i, 1)) = False
            'Rename Specialty categories
            If IsEmpty(Sheets("Claims").Cells(i, 6)) = True Then
                Sheets("Claims").Cells(i, 6).Value = "Unknown-Specialty"
            End If


            'Rename Specialty categories
            If IsEmpty(Sheets("Claims").Cells(i, 7)) = True Then
                Sheets("Claims").Cells(i, 7).Value = "Unknown-PS"
            End If


            'Rename Specialty categories
            If Sheets("Claims").Cells(i, 8).Value = "162+" Then
                Sheets("Claims").Cells(i, 8).Value = 162
            End If


            'Rename Specialty categories
            If IsEmpty(Sheets("Claims").Cells(i, 10)) = True Then
                Sheets("Claims").Cells(i, 10).Value = "Unknown-month"
            End If


            'Rename Specialty categories
            If IsEmpty(Sheets("Claims").Cells(i, 11)) = True Then
                Sheets("Claims").Cells(i, 11).Value = "Unknown-PCG"
            End If


            '*******************************************************
            'Rename CharlsonIndex categories

            If Sheets("Claims").Cells(i, 12).Value = "0" Then
                Sheets("Claims").Cells(i, 12).Value = "Charl-0"
            End If

            If Sheets("Claims").Cells(i, 12).Value = "1-2-2013" Then
                Sheets("Claims").Cells(i, 12).Value = "Charl-3-4"
            End If

            If Sheets("Claims").Cells(i, 12).Value = "3-4-2013" Then
                Sheets("Claims").Cells(i, 12).Value = "Charl-1-2"
            End If

            If Sheets("Claims").Cells(i, 12).Value = "5+" Then
                Sheets("Claims").Cells(i, 12).Value = "Charl-5+"
            End If


            '*******************************************************

            'Rename Specialty categories
            If IsEmpty(Sheets("Claims").Cells(i, 13)) = True Then
                Sheets("Claims").Cells(i, 13).Value = "Unknown-PG"
            End If


            i = i + 1
    Wend
End Sub
```

Transforming data

VBA code to transform categorical attributes into numerical attribute

```
Sub CategoToColumn() 'Transform categories into columns
   i = 2
```

```
j = 2
 While IsEmpty(Sheets("Claims").Cells(i, 1)) = False
     Sheets("EindData").Cells(j, 1).Value = Sheets("Claims").Cells(i, 1).Value

         While Sheets("EindData").Cells(j, 1).Value = Sheets("Claims").Cells(i, 1).Value

             n = 2
             While Sheets("EindData").Cells(1, n).Value <> Sheets("Claims").Cells(i, 6).Value And n <= 14
             n = n + 1
             Wend

             If Sheets("EindData").Cells(1, n).Value = Sheets("Claims").Cells(i, 6).Value Then
             Sheets("EindData").Cells(j, n).Value = Sheets("EindData").Cells(j, n).Value + 1
             End If



             n = 15
             While Sheets("EindData").Cells(1, n).Value <> Sheets("Claims").Cells(i, 7).Value And n <= 23
             n = n + 1
             Wend

             If Sheets("EindData").Cells(1, n).Value = Sheets("Claims").Cells(i, 7).Value Then
                 Sheets("EindData").Cells(j, n).Value = Sheets("EindData").Cells(j, n).Value + 1
             End If


             If Sheets("Claims").Cells(i, 8).Value >= 0 And Sheets("Claims").Cells(i, 8).Value <= 25 Then

                 Sheets("EindData").Cells(j, 24).Value = Sheets("EindData").Cells(j, 24).Value + 1
             End If

             If Sheets("Claims").Cells(i, 8).Value > 25 And Sheets("Claims").Cells(i, 8).Value <= 50 Then

                 Sheets("EindData").Cells(j, 25).Value = Sheets("EindData").Cells(j, 25).Value + 1
             End If

             If Sheets("Claims").Cells(i, 8).Value >= 51 And Sheets("Claims").Cells(i, 8).Value <= 75 Then

                 Sheets("EindData").Cells(j, 26).Value = Sheets("EindData").Cells(j, 26).Value + 1
             End If

             If Sheets("Claims").Cells(i, 8).Value >= 76 And Sheets("Claims").Cells(i, 8).Value <= 100 Then

                 Sheets("EindData").Cells(j, 27).Value = Sheets("EindData").Cells(j, 27).Value + 1
             End If

             If Sheets("Claims").Cells(i, 8).Value >= 101 And Sheets("Claims").Cells(i, 8).Value < 125 Then

                 Sheets("EindData").Cells(j, 28).Value = Sheets("EindData").Cells(j, 28).Value + 1
             End If

             If Sheets("Claims").Cells(i, 8).Value >= 126 And Sheets("Claims").Cells(i, 8).Value < 150 Then

                 Sheets("EindData").Cells(j, 29).Value = Sheets("EindData").Cells(j, 29).Value + 1
             End If

             If Sheets("Claims").Cells(i, 8).Value >= 151 And Sheets("Claims").Cells(i, 8).Value < 175 Then

                 Sheets("EindData").Cells(j, 30).Value = Sheets("EindData").Cells(j, 30).Value + 1
             End If


             n = 31
             While Sheets("EindData").Cells(1, n).Value <> Sheets("Claims").Cells(i, 10).Value And n <= 43
             n = n + 1
             Wend
             If Sheets("EindData").Cells(1, n).Value = Sheets("Claims").Cells(i, 10).Value Then
                 Sheets("EindData").Cells(j, n).Value = Sheets("EindData").Cells(j, n).Value + 1
             End If




             n = 44
             While Sheets("EindData").Cells(1, n).Value <> Sheets("Claims").Cells(i, 11).Value And n <= 89
             n = n + 1
             Wend

             If Sheets("EindData").Cells(1, n).Value = Sheets("Claims").Cells(i, 11).Value Then
                Sheets("EindData").Cells(j, n).Value = Sheets("EindData").Cells(j, n).Value + 1
             End If

             n = 90
             While Sheets("EindData").Cells(1, n).Value <> Sheets("Claims").Cells(i, 13).Value And n <= 107
             n = n + 1
             Wend

             If Sheets("EindData").Cells(1, n).Value = Sheets("Claims").Cells(i, 13).Value Then
                Sheets("EindData").Cells(j, n).Value = Sheets("EindData").Cells(j, n).Value + 1
             End If
```

```
                n = 108
                While Sheets("EindData").Cells(1, n).Value <> Sheets("Claims").Cells(i, 12).Value And n <= 111
                n = n + 1
                Wend

                If Sheets("EindData").Cells(1, n).Value = Sheets("Claims").Cells(i, 12).Value Then
                   Sheets("EindData").Cells(j, n).Value = Sheets("EindData").Cells(j, n).Value + 1
                End If
                i = i + 1
            Wend
            j = j + 1
    Wend
End Sub
```

Assigning the DaysInHospital

VBA code to assign, for each member/year, the corresponding DaysInHospital

```
Sub DaysInHospital() 'Assign, for each member/year, the corresponding DaysInHospital
          i = 2
          j = 2
    While IsEmpty(Sheets("EindData").Cells(i, 1)) = False

          While IsEmpty(Sheets("DaysInHospital").Cells(j, 1)) = False And Sheets("EindData").Cells(i, 1).Value <>
          Sheets("DaysInHospital").Cells(j, 1).Value
        j = j + 1
        Wend

        If Sheets("DaysInHospital").Cells(j, 1).Value = Sheets("EindData").Cells(i, 1).Value Then

                Sheets("EindData").Cells(i, 125).Value = Sheets("DaysInHospital").Cells(j, 3).Value
        End If
        i = i + 1
    Wend

    'Fill zeros in
          i = 2

    While IsEmpty(Sheets("EindData").Cells(i, 1)) = False
        For j = 2 To 124
            If Sheets("EindData").Cells(i, j).Value = "" Then

                  Sheets("EindData").Cells(i, j).Value = 0
            End If
        Next j
        i = i + 1
    Wend
End Sub
```

Assigning the age and sex category

```
Sub AgeSex()

i = 2
j = 2
    While IsEmpty(Sheets("EindData").Cells(i, 1)) = False


        While IsEmpty(Sheets("AgeSex").Cells(j, 1)) = False And Sheets("EindData").Cells(i, 1).Value <> Sheets("AgeSex").Cells(j,
1).Value
        j = j + 1
        Wend

        If Sheets("AgeSex").Cells(j, 1).Value = Sheets("EindData").Cells(i, 1).Value Then
                n = 112
                While IsEmpty(Sheets("EindData").Cells(1, n)) = False And Sheets("EindData").Cells(1, n).Value <>
Sheets("AgeSex").Cells(j, 2).Value
                n = n + 1
                Wend
                Sheets("EindData").Cells(i, n).Value = 1

                n = 122
                While IsEmpty(Sheets("EindData").Cells(1, n)) = False And Sheets("EindData").Cells(1, n).Value <>
Sheets("AgeSex").Cells(j, 3).Value
                n = n + 1
                Wend
                Sheets("EindData").Cells(i, n).Value = 1
        End If
        i = i + 1
    Wend
End Sub
```

Appendix C

Application of the RBF network model

Matlab code to train and test the RBF model for different number of kernels and different types of activation function.

```
%RBF without PCA model for different number of kernels and different types of activation function.

AcFun={'gaussian','tps','r4logr'};  %Activation functions
for i=1:3
   modefiedData=[Year11(:,1:124);Year12(:,1:124);Year21(:,1:124);Year22(:,1:124)];
   targetData=[Year11(:,125);Year12(:,125); Year21(:,125); Year22(:,125)];
    NUMBER_OF_ITERATIONS=100;
    V=length(Year22(:,1)); %total number of data
    fold=10;
    folds=mod(1:V,10)+1;         %generate "fold_id's" - integers 1-5
    folds=folds(randperm(V))';

    N=[2 3 4 5 7 10 15 25];
    Accuracy=zeros(8,1);
    for k=1:length(N)

        net = rbf(length(modefiedData(1,:)) , N(k), 1,AcFun{j});
        accuracy=zeros(fold,1);

        for i=1:fold
            %select train:
            x_train=modefiedData(folds~=i,:);
            t_train=targetData(folds~=i,:);
            %define test:
            x_test=modefiedData(folds==i,:);
            t_test=targetData(folds==i,:);
                options=foptions;
                options(14)=NUMBER_OF_ITERATIONS;
                net = rbftrain(net, options,  x_train, t_train);
                y = rbffwd(net, x_test);

                accuracy(i)=sqrt(sum((log(y+1)-log(t_test+1)).^2)/length(t_test));
        end
        Accuracy(k)=mean(accuracy);

    end
    figure,plot(N,Accuracy,'-.r');

    h = legend('Accuracy',1);
    set(h,'Interpreter','none');
    str= strcat('The Accuracy of the RBF network with 100 iterations and "', AcFun(j),'" activation function');
    title(str);
end
```

Application of the MLP network model

Matlab code to train and test the MLP model for different number of hidden units and different types of activation function for output.

```
%Accuracy of the  multilayer perceptron (MLP) model

   modefiedData=[Year11(:,1:124);Year12(:,1:124);Year21(:,1:124);Year22(:,1:124)];
   targetData=[Year11(:,125);Year12(:,125); Year21(:,125); Year22(:,125)];
    V=length(Year22(:,1)); %total number of data
    fold=10;
    folds=mod(1:V,10)+1;         %generate "fold_id's" - integers 1-5
    folds=folds(randperm(V))';

    N=[2 5 10 20 25];
    Accuracy=zeros(length(N),1);
  for k=1:length(N)

  net = mlp(length(modefiedData(1,:)),N(k) , 1, 'softmax');% possible output unit activation function: 'linear', 'logistic' and
'softmax'

       accuracy=zeros(fold,1);
        options = foptions;
        options(1) = -1;

        for i=1:fold
            %select train:
            x_train=modefiedData(folds~=i,:);
            t_train=targetData(folds~=i,:);
            %define test:
            x_test=modefiedData(folds==i,:);
            t_test=targetData(folds==i,:);
```

```
                [net, options,elog] = netopt(net, options, modefiedData, targetData, 'scg');
                y = mlpfwd(net, x_test);
                accuracy(i)=sqrt(sum((log(y+1)-log(t_test+1)).^2)/length(t_test));
        end
        Accuracy(k)=mean(accuracy);
  end
figure,plot(N,Accuracy,'-.r');

    h = legend('Accuracy',1);
    set(h,'Interpreter','none');
    str= strcat('The Accuracy of the MLP network with  "softmax" as an activation function');
    title(str);
```

Application of Multiple Lineair Regression model

Matlab code to train and test the Multiple Lineair Regression model.

```
%Accuracy of the Multiple linear regression model

   modefiedData=[Year11(:,1:124);Year12(:,1:124);Year21(:,1:124);Year22(:,1:124)];
   targetData=[Year11(:,125);Year12(:,125); Year21(:,125); Year22(:,125)];
    V=length(Year22(:,1)); %total number of data
    fold=10;
    folds=mod(1:V,10)+1;         %generate "fold_id's" - integers 1-5
    folds=folds(randperm(V))';


                accuracy=zeros(fold,1);


        for i=1:fold
            %select train:
            x_train=modefiedData(folds~=i,:);
            t_train=targetData(folds~=i,:);
            %define test:
            x_test=modefiedData(folds==i,:);
            t_test=targetData(folds==i,:);
            b = regress(targetData,modefiedData);
                y = x_test*b;

                accuracy(i)=abs(sqrt(sum((log(y+1)-log(t_test+1)).^2)/length(t_test)));
        end
  AccuracyLRM=mean(accuracy);
```

Applying PCA

Matlab code for applying the Principal Components Analysis (PCA) model for all the original datasets, which, due to the volume of the datasets I have split into two subsets for each year, namely Yer11 and Yer12 for year 1, Yer21 and Yer22 for year 2 and Yer31 and Yer32 for year 3.

```
%Using Principal Components Analysis (PCA) to reduce the number of dimensions of the original dataset
N=60 %number of PC
[PCCOEFF, PCVEC] = PCA(Year11(:,2:124),N);
encode=PCVEC'*Year11(:,2:124)';
Yer11=encode';

[PCCOEFF, PCVEC] = PCA(Year12(:,2:124),N);
encode=PCVEC'*Year12(:,2:124)';
Yer12=encode';

[PCCOEFF, PCVEC] = PCA(Year21(:,2:124),N);
encode=PCVEC'*Year21(:,2:124)';
Yer21=encode';

[PCCOEFF, PCVEC] = PCA(Year22(:,2:124),N);
encode=PCVEC'*Year22(:,2:124)';
Yer22=encode';

[PCCOEFF, PCVEC] = PCA(Year31(:,2:124),N);
encode=PCVEC'*Year31(:,2:124)';
Yer31=encode';

[PCCOEFF, PCVEC] = PCA(Year32(:,2:124),N);
encode=PCVEC'*Year32(:,2:124)';
Yer32=encode';
```

Applying the Classification Decision Tree model using PCA,

Matlab code for the predictive model

```
%Prediction model using the classification decision tree

modefiedData=[Year11(:,1:124);Year12(:,1:124);Year21(:,1:124);Year22(:,1:124)];
targetData=[Year11(:,125);Year12(:,125); Year21(:,125); Year22(:,125)];
TREE=ClassificationTree.fit(modefiedData,targetData);

y31=predict(TREE,Yer31(:,2:60));
y32=predict(TREE,Yer32(:,2:60));


TargetDTreePCA=[y31;y32];
```